《数字逻辑》综合设计指导书

湖南大学信息科学与工程学院

计算机工程系

数字逻辑课程教学组

1. 设计目的

连贯运用《数字逻辑》所学到的知识，熟练掌握EDA工具的使用方法，为学习好后续《计算机原理》课程做铺垫。

1. 设计任务

① 按给定的数据格式和指令系统，使用EDA工具设计一台用硬连线逻辑控制的简易计算机系统；

② 要求灵活运用各方面知识，使得所设计的计算机系统具有较佳的性能；

③ 对所做设计的性能指标进行分析，整理出设计报告。

1. 数据格式与指令系统

本设计的主要目的是希望学生巩固在《数字逻辑》课程中学到的理论知识，并加以灵活运用。因此，要设计的计算机系统非常简单。这个系统具有寄存器直接寻址和寄存器间接寻址两种寻址方式，除跳转指令为双字节指令外，其它指令均为单字节指令，字长为8位。

1. 数据格式

数据字采用8位二进制定点补码表示，其中最高位（第7位）为符号位，小数点可视为最左或最右，其数值表示范围分别为：-1≤X＜1或-128≤X＜127。

1. 寻址方式

指令的高4位为操作码，低4 位分别用2位表示目的寄存器和源寄存器的编号，或表示寻址方式。本机有2种寻址方式。

⑴ 寄存器直接寻址

当R1和R2均不是“11”时，R1和R2分别表示两个操作数所在寄存器的地址（寄存器编号），其中R1为目标寄存器地址，R2为源寄存器地址。

R1或R2的值 指定的寄存器

00 A寄存器

01 B寄存器

10 C寄存器

⑵ 寄存器间接寻址

当R1或R2中有一个为“11”时，表示相应操作数的地址在C寄存器中。

1. 指令系统

该机给定的指令系统有15条指令，指令格式见指令系统表。应该指出的是，各条指令的编码形式可以多种多样。为叙述方便，下面采用汇编符号对指令进行描述，其中R1和R2分别表示“目标”和“源”两个寄存器，M表示地址在寄存器C中的存贮单元。

|  |  |  |
| --- | --- | --- |
| 表1 指 令 系 统 表 | | |
| 指令的汇编符号 | 指令的功能 | 指令的二进制编码 |
| MOV R1，R2 | （R2）→ R1 | 0011 R1 R2 |
| MOV M，R2 | （R2）→（C） | 0011 11 R2 |
| MOV R1，M | （（C））→R1 | 0011 R1 11 |
| ADD R1，R2 | （R1）+（R2）→ R1 | 1001 R1 R2 |
| SUB R1，R2 | （R1）-（R2）→ R1 | 0110 R1 R2 |
| AND R1，R2 | （R1）∧（R2）→ R1 | 1110 R1 R2 |
| NOT R1 | /（R1）→ R1 | 0101 R1 XX |
| SHR R1 | （R1）逻辑右移一位→ R1 | 1010 R1 00 |
| SHL R1 | （R1）逻辑左移一位→ R1 | 1010 R1 11 |
| JMP add | add → PC | 0001 00 00，address |
| JZ add | 结果为0时add → PC | 0001 00 01，address |
| JC add | 结果有进位时add → PC | 0001 00 10，address |
| IN R1 | （开关7-0）→ R1 | 0010 R1 XX |
| OUT R1 | （R1）→ 发光二极管7-0 | 0100 R1 XX |
| NOP | （PC）+1 → PC | 0111 00 00 |
| HALT | 停机 | 1000 00 00 |

1. 数据通路及其说明

计算机的工作过程可以看作是许多不同的数据流和控制流在机器各部分之间的流动，数据流所经过的路程称作机器的数据通路。数据通路不同，指令执行所经过的操作过程就不同，机器的结构也就不一样。如何设计一个好的数据通路已经超出了本课程的范围，在此我们不予讨论。我们假设所设计的计算机的数据通路如图1所示。

1. 数据传送类指令的执行过程

寄存器之间的传送

MOV R1，R2

要求完成的操作为（R2）→R1，执行过程为：

⑴ 在RE的控制下，由R2所给定的编码选择RA1、RA0从通用寄存器组中读出R2的内容，并由Lod SA送入寄存器SA暂存；

⑵ 选择S3～S0和M，使SA中的内容经ALU送入总线BUS，再由WE控制和R1给定的编码选择WA1、WA0，将BUS上的数据写入通用寄存器R1。

寄存器与内存之间的传送

MOV M，R2

MOV R1，M

第一种情况表示要完成的操作为（R2）→（C），执行过程为：

⑴ 在RE的控制下，从C寄存器中读出内容送入MAR；

⑵ 在RE和R2编码的控制下，读出R2的内容，经SA和ALU送入BUS，然后在/CS和XL的配合下，写入MAR指定的内存单元。

第二种情况与第一种情况相似，只是数据传送的方向相反，即（（C））→R1，其基本操作可以类推。



图1 模型机数据通路

1. 算术逻辑运算类指令的执行过程

ADD R1，R2

SUB R1，R2

AND R1，R2

这类指令的执行过程为：

⑴ 在RE的控制下，由R1所给定的编码选择RA1、RA0从寄存器组中读出R1的内容，并由Lod SA送入SA暂存；

⑵ 在RE的控制下，由R2所给定的编码选择RA1、RA0从寄存器组中读出R2的内容，并由Lod SB和/R→SB控制送入寄存器SB暂存；

⑶ 对算术逻辑部件ALU发出相应的S3～S0和M选择信号，使其完成操作码所规定的操作，对两个暂存器中的数据进行运算，并在WE和WA1、WA0的控制下将运算结果送回R1。

1. 移位指令的执行过程

SHR R1

SHL R1

移出位送入进位触发器Cf，移入位补0。这类指令的执行过程为：

⑴ 在RE和R1的控制下，从R1指定的寄存器中取数，并将送入一个暂存器（SA或SB）中暂存；

⑵ 在S3～S0和M选择信号的控制下，使暂存器中的数据在通过ALU送入总线前完成向左移位或向右移位，再在WE和R1的控制下将总线上经过移位的数据写回到原来的寄存器中。

但是，标准的ALU模块没有移位功能，需要在该模块出口与总线接口处增加一部分电路以实现相应的移位功能（此问题请同学们自行解决）。

1. 转移类指令的执行过程

JMP add

JZ add

JC add

这类指令为双字节指令，第一字节为指令码，第二字节为转移目标地址。这类指令的执行过程为：

⑴ 在PC→MAR信号和Lod MAR信号的控制下，将程序计数器PC中的内容送入存储器地址寄存器MAR中，PC在LD PC和Lod PC的控制下加1；

⑵ 如果条件满足，则在DL和/CS信号控制下，将由MAR所指定存储单元的内容送入BUS，然后在LD PC和Lod PC的配合下，将BUS上的内容写入PC。如果条件不满足，则不执行其它任何操作，立即执行下一条新的指令。

当数据通路设计好之后，就要进行详细电路设计，这时需要考虑其它各种因素，比如进行触发器Cf的设置，符号位的处理，溢出的判断，等等。

1. 控制器设计

有了指令系统和数据通路之后，就可以进入控制器设计阶段。控制器设计有两种方法，一种是组合逻辑实现方法，另一种是微程序实现方法。我们采用第一种方法。

1、微控制信号

指令寄存器IR接收到一条机器指令后，这条指令就被译码执行。指令通过译码产生出的各种控制信号在时钟信号的配合下控制着指令执行的全过程。为此，需要将执行每条指令所需的全部基本微操作的控制信号罗列出来，进行综合分析、化简，并落实到不同的周期、节拍之中，然后用各种逻辑门电路实现。以下是所用控制信号列表。

|  |  |  |
| --- | --- | --- |
| 表2 控制信号及功能表 | | |
| 序号 | 信号 | 功能 |
| 1 | “结果”为0跳 | 当是指令“JZ add”时，此位为“1”。 |
| 2 | Cf=1跳 | 当是指令“JC add”时，此位为“1”。 |
| 3 | 1→Cf | 当是可能产生进位而此进位必须送到进位触发器Cf时，此位为“1”。 |
| 4 | HALT | 此位为“1”时停机，下次输入人工操作。 |
| 5 | F→BUS | ALU的运算结果通过移位门直接送到总线BUS的对应位。 |
| 6 | FL→BUS | ALU的运算结果通过移位门左移一位送到总线BUS，且F7送Cf。 |
| 7 | FR→BUS | ALU的运算结果通过移位门右移一位送到总线BUS，且F0送Cf。 |
| 8 | Lod SA | 上升沿有效，把BUS上的数据打入暂存器A。 |
| 9 | Lod SB | 上升沿有效，把BUS上的数据打入暂存器B。 |
| 10 | Lod PC | 上升沿有效，当LD PC=1时把BUS上的数据打入程序计数PC，否则对PC加1计数。 |
| 11 | LD PC | 与Lod PC配合使用的对PC进行控制的电平信号。 |
| 12 | Lod MAR | 上升沿有效，把BUS上的数据打入地址寄存器MAR。 |
| 13 | Lod IR | 上升沿有效，把BUS上的数据打入指令寄存器IR。 |
| 14 | PC→MAR  /R→MAR | 为“1”时将PC中的内容传送到地址寄存器MAR，否则将通用寄存器中的内容传送到地址寄存器MAR。 |
| 15 | /RE | 对通用寄存器组允许读，低电平有效。 |
| 16 | /WE | 对通用寄存器组允许写，低电平有效。 |
| 17 | DL | 对存储器RAM允许读。 |
| 18 | XL | 对存储器RAM允许写。 |
| 9 | /CS | 对存储器允许访问，低电平有效。 |
| 20 | 1→M | 使M=1，表示ALU进行逻辑运算操作。 |
| 21 | S3～S0 | 使ALU执行各种运算的控制位。 |
| 22 | IR3:2→RA1:0 | 当通用寄存器组的访问由指令的第3、2两位确定时，该位为“1”。 |
| 23 | IR1:0→RA1:0 | 当通用寄存器组的访问由指令的第1、0两位确定时，该位为“1”。 |

2、指令周期与工作脉冲设置

指令同期与数据通路结构、指令执行方式有关。指令可以串行执行，也可以并行执行。本设计采用串行工作方式，即“读取—执行—再读取—再执行……”。串行工作方式虽然工作速度和主机效率都要差一些，但它的控制简单。因此，本机指令周期可以确定为：



读取指令的时间随所使用的RAM的性能而异。执行一条指令所需工作脉冲的个数与宽度要依据控制流和数据流所经过的路径与各级门的最大延迟而定。例如，本机中写入RAM和寄存器组的操作显然不能发生在“执行阶段”的任意时刻，它必须是在运算结果已经产生，并被传送到总线的适当时刻才能“写”，这就需要工作脉冲来控制定序。需要工作脉冲定序的其它微操作还有：Lod SA、Lod SB、Lod MAR、Lod IR和Lod PC等等。

1. 控制台与控制指令设计

有了数据通路和控制器，还要有输入/输出部分，我们才能将编好的程序送入系统，看到运算结果。为简单起见，我们仅考虑最简单的I/O部分。虽然是最简单的，但仍必须有以下几个功能：

① 能够启动系统运行；

② 能够输入地址和程序；

③ 能够检查内存的内容；

④ 能够终止系统的运行；

⑤ 能够对系统进行初始化；

⑥ 能够提供某些出错信息，如溢出等。

1. 命令键设置及实现

为了实现控制台的功能，我们可以设置以下几个命令键：

⑴ START键：启动系统运行，起始地址由PC当前内容确定；

⑵ STOP键：终止系统运行；

⑶ INA键：输入地址，将开关上设置的数作为地址送入PC中；

⑷ INP键：输入程序，将开关上设置的数作为机器指令代码送入当前PC所指定的内存单元中；

⑸ CHP键：检查程序，将当前PC所指定的内存单元的内容送显示器显示；

⑹ RESTART键：系统总复位命令。

读者还可以设置一些其它命令键。在上面6个键中，INA、INP、CHP不仅要启动系统运行，还需要在命令执行完后停机。要注意，这样的命令没有自己的指令格式，需要另外设计编码电路对这些命令进行编码，并将其送入指令寄存器IR。

1. 输入输出部件

输入输出可以采用二进制、十进制或ASCII码。设备有多种多样，如LCD显示器及键盘、打印机等。为简单起见，本机采用最简单的二进制开关作为输入部件，发光二极管作为输出部件。发光二极管所显示的内容要由寄存器保存。